Theoretical Computer Science 412 (2011) 3738–3751

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Approximation algorithms for grooming in optical network design

Spyridon Antonakopoulos *, Lisa Zhang

Bell Laboratories, Alcatel-Lucent, Murray Hill, NJ, USA

ARTICLE INFO

Article history:
Received 11 January 2010
Received in revised form 21 March 2011
Accepted 29 March 2011
Communicated by D. Peleg

Keywords:
Traffic grooming
Optical network
Approximation algorithms
Cycle packing

ABSTRACT

We study traffic grooming in optical network design, where the goal is to aggregate low-bandwidth traffic streams to utilize efficiently high-bandwidth media such as wavelength channels. More precisely, given traffic demands to be routed in a network, the design problem is to define a collection of *light paths* such that each demand can follow a sequence of consecutive light paths. Each light path has a unit-wavelength bandwidth, and multiple sub-wavelength demands may share a common light path. Traffic must enter and depart from a light path at its two endpoints only. Most previous work on grooming focused on the ring topology and typically involved only uniform bandwidth demands, whereas we deal with more general settings.

Two objectives are considered. One is to minimize the total cost of equipment necessary to support the light paths; the other is simply to minimize the number of light paths. Even for the extremely restricted special case of a line topology and traffic demands that request half wavelength bandwidth, we show that both objectives are APX-hard to optimize, which means we cannot approximate the optimum arbitrarily closely. On the other extreme of generality, for arbitrary network topologies and traffic demands that request arbitrary amounts of bandwidth, we show a logarithmic approximation for cost minimization and a 2-approximation for minimizing light path counts.

Furthermore, we discover that the special case of half-wavelength demands has rich combinatorial properties, closely related to graph problems such as cycle packing and pattern matching problems such as interchange distance in strings. We show how to approximate both objectives up to small constant factors in this case, while similarly improving the approximation and hardness of the interchange distance problem as well.

1. Introduction

In modern optical networks with Dense Wavelength-Division Multiplexing (DWDM) technology, data can be transported via hundreds of channels, on distinct wavelengths, which are carried over a single fiber. Nevertheless, the bandwidth requirement of an individual traffic demand is often much smaller than the capacity of a wavelength channel. Therefore, to make efficient use of the available bandwidth, several low-rate demands may be bundled together into a shared wavelength, in a process known as *traffic grooming*. In effect, this constitutes another form of multiplexing that takes place before DWDM.

Earlier work on grooming has created a sizable body of literature that focuses on the ring topology, since it was motivated by the SONET/SDH technology over rings. As an example, 16 traffic streams of bitrate OC-3 can be multiplexed into one stream of higher bitrate OC-48 using devices such as the SONET Add/Drop Multiplexers, or SADMs, where the number 16 is called the *grooming factor*. More recent work on grooming is migrating towards general mesh topologies and demands that request an arbitrary sub-wavelength bandwidth. This is primarily motivated by applications like packet over WDM. In that

* Corresponding author. Tel.: +1 908 743 3305.
E-mail addresses: spyros@research.bell-labs.com, spyros.antonakopoulos@gmail.com (S. Antonakopoulos), ylz@research.bell-labs.com (L. Zhang).

0304-3975/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2011.03.034

setting, devices such as packet routers perform the function of traffic grooming, and groomed traffic is then carried over the WDM network in integral units of wavelength channel capacity.

In a solution to the grooming problem, each packet stream traverses a sequence of *light paths*. A light path is realized by allocating the same wavelength channel along a sequence of physical fiber links in the WDM network, and hence its capacity is equal to that of the channel. Each endpoint of a light path is equipped with a packet router, which is connected to a co-located optical add/drop multiplexer. We assume that packet streams are dropped from the optical domain at the end of the one light path and are regroomed with other streams at the beginning of the next light path. (If no regrooming takes place, then the two light paths can be concatenated into one for an improved solution.) We also assume that injecting traffic into or extracting traffic from a light path only happens at its endpoints. Our assumptions are based on the capabilities of state-of-the-art commercially available packet and optical components, such as the Alcatel-Lucent 1850 Universal Packet Multiplexer.

Solving the grooming problem consists of two interconnected parts: (i) designing light paths, which includes specifying the physical route of each path; and (ii) assigning each packet stream to a sequence of light paths. The cost of a solution is the total cost of equipment installed at each endpoint of the light paths, as well as the cost of supporting the wavelength channel that realizes each light path along the fiber links. Hence, one natural objective is cost minimization subject to feasibility.

We should point out that grooming is not an isolated problem for optical network design. An optical backbone handles small sub-wavelength traffic streams as well as larger streams in integer multiples of wavelengths. Grooming that aggregates sub-wavelength streams into wavelength streams can be viewed as a step preceding the WDM-only phase of the design, where wavelength streams are multiplexed using WDM technology. Nevertheless, we do not consider this latter phase herein, as it has generated a huge amount of research interest by itself, including topics such as routing and wavelength assignment, buy-at-bulk network design, protection and resilience, etc. We concentrate on traffic grooming only in this paper.

1.1. Problem statement

In the formal definition of the problem, the input is an undirected connected graph $G = (V, E)$, a non-negative cost function $c : E \to \mathbb{N}$, and a collection of traffic demands $\mathcal{D}$. The graph G represents the WDM network topology, where the node set V represents the locations for packet and WDM equipment, and the edge set E represents the optical fiber connections. Moreover, $c(e)$ is the cost of supporting one light path, or one wavelength, over edge e. Each demand $d \in \mathcal{D}$ is specified by a pair of *endpoints* $(u, v) \in \binom{V}{2}$, and requests an arbitrary bitrate $b_d \in \mathbb{N}^*$. Note that multiple demands with the same pair of endpoints are allowed in $\mathcal{D}$. The undirected multigraph $G^d = (V, \mathcal{D})$ is called the *demand graph* of the grooming instance, where each edge (u, v) corresponds to a demand with endpoints u and v. We also use B to denote the bitrate of a single wavelength channel.

The output is a collection of light paths $\mathcal{P}$, each light path being represented as a *physical route* (i.e. a path, in the graph-theoretic sense) in G, as well as a binary relation R over $\mathcal{P}$ and $\mathcal{D}$, where pRd means that light path p serves demand d. We define the *light path graph* $G^p = (V, \mathcal{P})$ of the solution analogously to the demand graph, and denote by $\mathcal{P}_{uv}$ the subset of light paths with endpoints (u, v). R must be such that: (a) for any (u, v), the light paths in $\mathcal{P}_{uv}$ serve demands with total requested bitrate at most $B \cdot |\mathcal{P}_{uv}|$ (which implies that packet traffic may be flexibly distributed among functionally equivalent light paths); and (b) for any $d \in \mathcal{D}$, the light paths that serve d form a path in G^p with the same endpoints as d.

We express the total grooming cost as

$$|\mathcal{P}| + \sum_{e \in E} c(e)p(e), \tag{1}$$

where $p(e)$ is the number of light paths in $\mathcal{P}$ that use edge e. The first term of (1) is the number of light paths, which indicates the normalized *endpoint equipment cost*, whereas the second is the *wavelength usage cost* of the solution. This cost is edge-dependent, determined by factors such as the number of signal regenerations needed along the edge.

The sample instance in Fig. 1 has two demands d_1 and d_2 in a Y-shaped network, each requesting bandwidth $b_{d_1} = b_{d_2} \leq B/2$. One feasible solution (leftmost) deploys one end-to-end light path for each demand, so $|\mathcal{P}| = 2$ here. Another feasible solution (second from left) deploys one light path along the common route of d_1 and d_2, and one light path for each branch of d_1 and d_2. In this case, $|\mathcal{P}| = 3$. The third solution is feasible but suboptimal, because the two light paths for d_2 can be concatenated into one. The solution on the right is not feasible, since traffic for d_2 cannot enter a light path except at its endpoints.

Grooming is closely related to a well-studied network design problem called *buy-at-bulk*. Buy-at-bulk takes as input an undirected graph $H = (V_H, E_H)$, a cost function $f : (E_H, \mathbb{Q}^+) \to \mathbb{Q}^+$ and a set of demands $\mathcal{D}$; it outputs a solution that specifies a routing path for each demand. The objective is to minimize the cost of a solution

$$\sum_e f(e, \ell(e)),$$

where $\ell(e)$ is the total bitrate of demands routed through e. Moreover, f is *subadditive*: $f(e, \ell_1) + f(e, \ell_2) \geq f(e, \ell_1 + \ell_2)$. That is, carrying bandwidths ℓ_1 and ℓ_2 collectively is more economical than carrying each one separately. This gives rise to the name "buy-at-bulk".

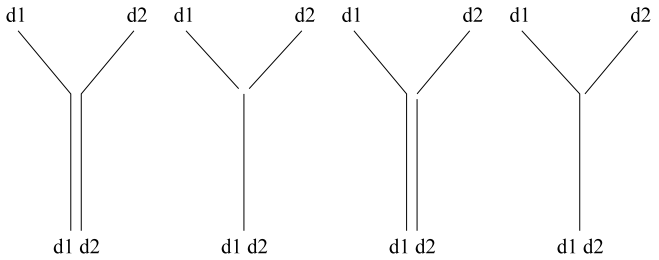

Fig. 1. Candidate solutions for a sample grooming instance with two demands. The lines drawn represent light paths.

In this paper we study two objective functions for grooming. The first is to minimize (1) and we refer to this problem as *grooming with general costs*. The second is to minimize only the first term of (1), i.e. $|\mathcal{P}|$. We refer to this problem as *light path minimization*.

We note that minimizing the second term of (1) alone, $\sum_{e\in E} c(e)p(e)$, is identical to a special case of buy-at-bulk network design. Indeed, since the light path count is not a consideration, we may assume that each light path spans just a single fiber link, in other words a single edge. Therefore, for a given routing of the demands, wavelength usage cost is minimized by creating the least possible number of light paths on each edge $e \in E$. In particular, if $\ell(e)$ is the total bitrate of demands routed through e, then $p(e) = \lceil \ell(e)/B \rceil$. In the buy-at-bulk literature, it is called the *single cable* case, referring to the uniform cable capacity of B and fixed per-cable cost $c(e)$ over e.

The first objective of general cost minimization is also related to buy-at-bulk, though the relationship is less straightforward, as we shall see later. The second objective of light path minimization is interesting for a number of reasons. To begin with, the endpoint equipment cost may be the dominating term in (1). Further, as mentioned earlier, grooming is not an isolated problem. For an optical backbone that carries both sub-wavelength packet traffic and wavelength demands, a step-by-step design approach would be to begin by solving the light path minimization problem on the sub-wavelength demands and then to treat each light path in the resulting solution as a unit-wavelength demand, together with other wavelength demands, in solving the WDM-only problem. This approach has the advantage of choosing the physical route of each light path as part of a "global" optimization of all wavelength demands.

In addition to proposing algorithms for general cases in which network topologies are arbitrary and demand bitrates vary arbitrarily, we also investigate special topologies (such as lines and trees) with *half-wavelength demands*, i.e. $b_d = B/2$ for all d. Observe that demand routes are fixed in these topologies. We are motivated to investigate if such extremely restricted cases are polynomially solvable and, in particular, whether fixed routing makes grooming easy. Note that fixed routing renders the buy-at-bulk network design problem trivial. However, somewhat surprisingly, not only is grooming half-wavelength demands on a line NP-hard, it is also APX-hard. This implies the existence of a constant $\epsilon > 0$ such that no polynomial-time algorithm can guarantee a solution that is within a factor of $1 + \epsilon$ of the optimal. Therefore, grooming half-wavelength traffic on a line cannot be approximated arbitrarily close to optimality.

On the positive side, studying these special cases reveals interesting combinatorial properties and close connections to graph-theoretic problems such as the *cycle packing* and *interchange distance* problems, as we shall see in later sections. Cycle packing seeks the largest set of edge-disjoint cycles in a given (multi)graph. The interchange distance problem aims to minimize the number of pair-wise swaps that would make two strings identical. Both problems arise frequently in computational biology.

Another dimension of our work is concerned with grooming under the *wavelength-parsimonious constraint*, which forces $p(e) = \lceil \ell(e)/B \rceil$ on every edge e. As implied by the examples in Fig. 1, an optimal grooming solution for either objective might not satisfy this constraint. Interestingly, however, comparing two solutions for light path minimization, one respecting the parsimonious constraint and the other not, guarantees better approximation ratios for grooming with general costs.

1.2. Related work

Grooming is a ubiquitous problem. Surveys of the extensive literature can be found, among others, in [1,2]. Numerous heuristics have been proposed for a large variety of settings, including instances with all-to-all, multicasting, or anycasting demands.

From this sizable body of literature, we make special mention of a line of work [3–6] dealing with grooming in SONET rings. In that framework each demand uniformly requests a fraction $1/K$ of the wavelength bandwidth, where K is the grooming factor. Under certain assumptions specific to SONET Unidirectional Path-Switched Ring topologies, optimizing the number of required SADMs is reduced to partitioning a graph into edge-disjoint subgraphs with up to K edges each, such that the sum of vertex cardinalities of these subgraphs is minimized.

Approximation algorithms for the latter problem were developed in [4–6]. Notably, wavelength usage was also considered as a (secondary) criterion in comparing the performance of those algorithms. It is worth remarking that certain approaches [7,8], pertaining to the special case where wavelength capacity restrictions are removed, resemble to some extent our techniques for grooming instances with half-wavelength demands, even though they differ substantially in their

Table 1
Approximation and inapproximability ratios for grooming.

Light path minimization	Approximation	Hardness
1/2-wavelength demands	1.305	$1+\epsilon$
Arbitrary demands	2	$1+\epsilon$
Grooming with general costs	**Approximation**	**Hardness**
1/2-wavelength demands	2	$1+\epsilon$
-line	1.590	$1+\epsilon$
-tree	1.812	$1+\epsilon$
Arbitrary demands	$O(\log n)$	$\Omega\left(\log^{1/4-\epsilon} n\right)$

details. Moreover, [9] presented a $(K+1)$-approximation algorithm for the dual problem of maximizing a SONET network's throughput given a limited number of SADMs at each node. A variant of grooming on path, star and tree networks was also studied in [10,11].

In our setting, which is somewhat different from the one above, it turns out that grooming is closely related with several well-known problems in combinatorial optimization. The most prominent of these is buy-at-bulk network design, as we have already remarked. Notable results most relevant to our grooming problem, among the many that we have already mentioned in previous sections, are the $O(\log n)$-approximation algorithm of [12] and the $\Omega\left(\log^{1/4-\epsilon} n\right)$-hardness result from [13,14].

Another closely related problem is cycle packing, i.e. determining the largest set of edge-disjoint cycles in a given (multi)graph. Possibly the earliest mention of this problem was by [15], in the context of sorting permutations by reversals. Since then, it has received substantial attention, in both its directed and undirected versions. Refer to [16–18] for approximation algorithms and [16,18,19] for inapproximability results.

Last but not least, the interchange distance problem is interconnected with both grooming and cycle packing. The study of interchange distance between strings was initiated by [20] in the 19th century, who focused on the special case of permutation strings. Recently, [21] revisited the problem from a computational perspective, as part of a broader class of rearrangement distances. Later, a $\frac{3}{2}$-approximation algorithm for interchange distance on general strings was proposed by [22], and the problem was also shown to be NP-hard.

1.3. Results

Table 1 summarizes our main approximation and hardness results on various aspects of grooming. The rest of the paper is organized as follows.

- In Section 2 we explore the close connection between grooming with general costs and the buy-at-bulk network design problem. We show a reduction that allows us to translate the logarithmic approximation ratio and polylogarithmic inapproximability ratio from buy-at-bulk to grooming with general costs.
- In Section 3 we concentrate on light path minimization. On arbitrary network topologies, we show a 2-approximation algorithm for demands with arbitrary b_d. In the very simple special case of half-wavelength demands on a line topology, we show that light path minimization is APX-hard. However, we also show an improved approximation ratio of 1.305 for half-wavelength demands on arbitrary topologies. This is accomplished by exploring a connection to the cycle packing problem and using a so-called *factor-revealing linear program*. Additionally, our results for half-wavelength demands directly improve the best-known approximation and hardness results for the interchange distance problem.
- In Section 4 we show small constant approximation ratios for grooming with general costs for half-wavelength demands on line and tree topologies. This is achieved by studying the wavelength-parsimonious variant of light path minimization. Combining algorithms for this variant with the one for the original light path minimization problem yields an improved approximation for grooming with general costs.
- Finally, Section 5 contains our conclusions, as well as directions of future work.

2. Grooming with general costs

Grooming with general costs is essentially equivalent to the single-cable buy-at-bulk network design.

Theorem 2.1. *Grooming with general costs on arbitrary topologies and with arbitrary demands is approximable within ratio $O(\log n)$, but hard to approximate within $\Omega\left(\log^{1/4-\epsilon} n\right)$ for any $\epsilon > 0$, where n is the number of nodes in the network G.*

Proof. We begin with an observation. Without loss of generality, the physical route of a light path with endpoints u and v should follow a u–v path in the network that is shortest with respect to the cost function $c(e)$. This holds because traffic using this light path can only be added or dropped at u and v.

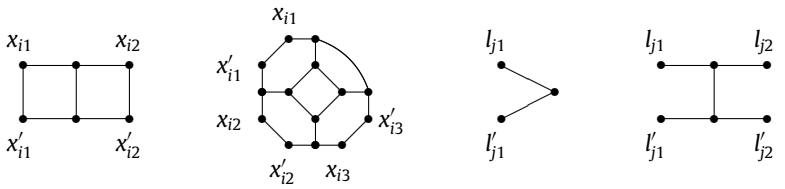

Fig. 2. Gadgets for the reduction from Max 2SAT-3 to light path minimization.

Given a grooming instance on network $G = (V, E)$, with cost function c and demand set $\mathcal{D}$, we create a single-cable buy-at-bulk instance on a network $H = (V, E_H)$, with cost function f and the same set of demands $\mathcal{D}$. The network H is a complete undirected graph defined on the node set V. Each cable has capacity B, and the cost $f(uv, \ell)$ of supporting bandwidth ℓ along link $uv \in E_H$ is

$$f(uv, \ell) = \left(1 + \sum_{e \in P_{uv}} c(e)\right) \cdot \left\lceil \frac{\ell}{B} \right\rceil,$$

where the term 1 reflects the normalized endpoint equipment cost and P_{uv} is a shortest path between u and v in G with respect to $c(e)$.

Based on the observation above, it is easy to see a one-to-one correspondence between a solution to the grooming instance and one to the buy-at-bulk instance, such that the two solutions have identical cost. Hence, the $O(\log n)$-approximation algorithm from [12] also gives a $O(\log n)$-approximation for grooming with general costs.

Regarding hardness, it suffices to show that the special case of grooming in which the second term $\sum_e c(e)p(e)$ of (1) overwhelmingly dominates the first term $|\mathcal{P}|$ is hard to approximate. As discussed in Section 1, when light path count is not a consideration, each light path spans a single fiber link. Consequently, $p(e) = \lceil \ell(e)/B \rceil$, where the load $\ell(e)$ is the total bitrate of demands going through e. This is equivalent to the single-cable buy-at-bulk problem, for which the $\Omega\left(\log^{1/4-\epsilon} n\right)$-hardness results of [13,14] hold. □

3. Grooming for light path minimization

3.1. Complexity results

We begin by considering the simplest possible case, namely grooming on a line topology where every demand requests a bitrate equal to $B/2$. Even such a very special case is not polynomially solvable. Consequently, we deduce that:

Theorem 3.1. *Grooming for light path minimization is* APX*-hard, even when restricted to instances with line topologies and half-wavelength demands.*

Proof. The theorem is established via an approximation-preserving reduction from Max 2SAT-3, which is APX-hard [23,24]. The APX-hardness of Max 2SAT-3 means that there exists a constant $\epsilon > 0$ such that, given a Max 2SAT-3 instance ϕ with n clauses, it is NP-hard to distinguish whether at most $c \geq n/2$ clauses of ϕ are satisfiable, or at least $(1 + \epsilon)c$ clauses are satisfiable. No fewer than half the clauses of ϕ are always satisfiable, e.g. by a simple greedy algorithm, hence the restriction $c \geq n/2$. In fact, if Y_1 and Y_2 are the numbers of clauses in ϕ with one and two literals, respectively, then the above restriction is strengthened to $c \geq Y_1/2 + 3Y_2/4$.

Our reduction is an adaptation of the reduction from Max 2SAT-3 to the undirected cycle packing problem, which was used to establish the APX-hardness of the latter [16]. Note that in a Max 2SAT-3 instance it may be assumed, without loss of generality, that every variable appears both in negated and in non-negated form, implying at the same time that it appears at most twice in each form. This assumption permits us to use simpler gadgets than [16].

Additionally, observe that the demand graph alone suffices to define an instance of light path minimization, provided that the underlying network is connected. Without loss of generality, then, we assume that the network topology of the instance we create for our reduction is simply a line, and henceforth focus on specifying its demand graph.

More specifically, using the gadgets shown in Fig. 2, we construct the demand graph G^d from a given Max 2SAT-3 instance, i.e. a 2SAT-3 formula ϕ that is subject to the assumptions we stated earlier. Either the leftmost or the middle-left gadget is employed to represent a variable x_i, depending on whether it appears twice or three times, respectively, in ϕ. Likewise, for each clause C_j of ϕ, we use either the middle-right or the rightmost gadget, depending on whether $C_j \equiv l_{j1}$ (one-literal clause) or $C_j \equiv l_{j1} \vee l_{j2}$ (two-literal clause), respectively.

Moreover, if $l_{j\lambda} \equiv x_i$ or $l_{j\lambda} \equiv \bar{x}_i$, where $\lambda \in \{1, 2\}$, then we identify node $x_{i\chi}$ with node $l_{j\lambda}$ and $x'_{i\chi}$ with $l'_{j\lambda}$, for some $\chi \in \{1, 2, 3\}$ of our choice; accordingly, we say that $l_{j\lambda}$ is *assigned to slot* χ *of* x_i. Each literal must be assigned to a distinct slot, and syntactically identical literals cannot be assigned to consecutive slots of the corresponding variable.

Next, we claim that if a truth assignment satisfies $\geq s$ clauses in ϕ, then there are $\geq X_2 + 2X_3 + s$ edge-disjoint cycles contained in G^d, and vice versa, where X_2 and X_3 are the numbers of variables that appear twice and three times in ϕ,

respectively. Combining this with Lemma 3.3 and Corollary 3.4 (in Section 3.2), we deduce that if a truth assignment satisfies $\geq s$ clauses of ϕ, then the number of light paths required is at most

$$|\mathcal{D}| - (X_2 + 2X_3 + s) = 2Y_1 + 5Y_2 + 6X_2 + 16X_3 - s,$$

and vice versa.

Indeed, given a hypothetical truth assignment that satisfies $\geq s$ clauses of ϕ, we present a collection Γ of $X_2 + 2X_3 + s$ edge-disjoint cycles in G^d. For each literal $l_{j\lambda}$ that evaluates to "false" and is assigned to, say, slot χ of some variable x_i, put in Γ the shortest possible cycle that includes edge $\left(x_{i\chi}, x'_{i\chi}\right)$ and is entirely contained in the gadget corresponding to x_i. If variable x_i appears three times in ϕ and the literal assigned to slot 2 of x_i evaluates to "false", then also add to Γ the 4-cycle on the upper-right corner of the gadget corresponding to x_i.

Furthermore, for each clause C_j that is satisfied, pick one of its literals $l_{j\lambda}$ that evaluates to "true" and suppose that it is assigned to slot χ of some variable x_i. Then, put in Γ the shortest possible cycle that includes edge $\left(x_{i\chi}, x'_{i\chi}\right)$ and all its other edges belong to the gadget corresponding to C_j. Overall, Γ contains one cycle for each variable that appears twice in ϕ, two cycles for each variable that appears three times, and one cycle for each satisfied clause. Thus, one direction of our claim is established. The proof of the opposite direction is essentially identical to the corresponding part of the proof of the reduction in [16].

Finally, recall that $Y_1 + 2Y_2 = 2X_2 + 3X_3 = |\phi|$, and that $c \geq Y_1/2 + 3Y_2/4$. Consequently, if the minimum number of light paths could be approximated within a

$$\begin{aligned}\frac{2Y_1 + 5Y_2 + 6X_2 + 16X_3 - c}{2Y_1 + 5Y_2 + 6X_2 + 16X_3 - (1+\epsilon)c} &\geq 1 + \frac{\epsilon \cdot 3Y_2/4}{47Y_2/3 - (1+\epsilon)3Y_2/4} \\ &= 1 + \frac{\epsilon}{188/9 - (1+\epsilon)} \\ &= 1 + \frac{\epsilon}{179/9 - \epsilon}\end{aligned}$$

factor, then we would be able to distinguish whether at least $(1+\epsilon)c$ clauses are satisfiable or at most c are, which is NP-hard. We hence conclude that grooming for light path minimization is APX-hard. □

Corollary 3.2. *Grooming with general costs is* APX*-hard, even when restricted to instances with line topologies and half-wavelength demands.*

3.2. Half-wavelength demands

Now, let us examine instances with half-wavelength demands on arbitrary topologies. Although quite restrictive, this provides valuable intuition for grooming and brings to light some interesting connections with other graph-theoretic problems. The lemma below is crucial in that regard.

Lemma 3.3. *Let $\mathcal{D}' \subseteq \mathcal{D}$ be a subset of demands, with cardinality m. Let $S(\mathcal{D}')$ be an optimal solution for $\mathcal{D}'$ with respect to the objective of light path minimization.*

1. *If $S(\mathcal{D}')$ requires fewer than $m-1$ light paths, then $\mathcal{D}'$ can be partitioned into non-empty disjoint subsets $\mathcal{D}'_1$ and $\mathcal{D}'_2$ such that each light path in $S(\mathcal{D}')$ serves demands in only one of $\mathcal{D}'_1$, $\mathcal{D}'_2$.*
2. *If the demand graph $G^{d'} = (V, \mathcal{D}')$ is acyclic, then we can efficiently construct a solution for $\mathcal{D}'$ with m light paths, and this is optimal.*
3. *If the demand graph $G^{d'} = (V, \mathcal{D}')$ is an m-cycle, $m \geq 2$, then we can efficiently construct a solution for $\mathcal{D}'$ with $m-1$ light paths, and this is optimal.*

Proof. Since every demand requests bitrate $b_d = B/2$, without loss of generality we can modify the binary relation R in the solution $S(\mathcal{D}')$ such that each light path serves at most two demands. We also say that two such demands *share* the light path. In a graph $\mathcal{G}$ expressing this *sharing relation* over $\mathcal{D}'$, every edge must correspond to a distinct light path. If $S(\mathcal{D}')$ uses fewer than $m-1$ light paths, then $\mathcal{G}$ has m vertices but fewer than $m-1$ edges, so it is not connected. As a result, $\mathcal{D}'$ can be partitioned into two or more mutually non-sharing subsets, which correspond to the vertex sets of the connected components of $\mathcal{G}$.

Suppose that $G^{d'} = (V, \mathcal{D}')$ is acyclic and has $\kappa \geq 1$ connected components. The light path graph $G^{p'} = (V, \mathcal{P}')$ determined by $S(\mathcal{D}')$ must have $\kappa' \leq \kappa$ components. Due to the acyclicity of $(V, \mathcal{D}')$, $|V| = m + \kappa$. Therefore, $(V, \mathcal{P}')$ should contain at least $|V| - \kappa' \geq m$ light paths. Of course, we never need more than m light paths: simply use one light path to serve each demand exclusively.

Finally, if $G^{d'}$ is an m-cycle, then let $(u_1, u_2), (u_2, u_3), \ldots, (u_{m-1}, u_m)$, and (u_m, u_1) be these m demands. We create $m-1$ light paths such that each of the first $m-1$ demands goes through exactly one light path, and the last demand goes through these $m-1$ light paths in order. This shows that $m-1$ light paths are enough. If fewer than $m-1$ light paths should suffice,

Algorithm 3.1 Grooming for light path minimization with half-wavelength demands

while the current demand graph is not acyclic **do**
  $C_{\text{short}} \leftarrow$ the shortest cycle in the demand graph
  use $|C_{\text{short}}| - 1$ light paths to serve the demands in C_{short}
  remove C_{short} from the demand graph
use one light path to serve each remaining demand

we would be able to partition $\mathcal{D}'$ into two mutually non-sharing subsets $\mathcal{D}'_1$ and $\mathcal{D}'_2$, as proven earlier. However, $(V, \mathcal{D}'_1)$ and $(V, \mathcal{D}'_2)$ would be acyclic. By the argument of the previous paragraph, they would require $|\mathcal{D}'_1|$ and $|\mathcal{D}'_2|$ light paths, respectively, for a total of $|\mathcal{D}'_1| + |\mathcal{D}'_2| = m$ light paths. Consequently, we have reached a contradiction. □

Corollary 3.4. *The optimal solution to the grooming instance uses exactly* $\mathsf{OPT} = D - \nu$ *light paths, where* $D = |\mathcal{D}|$ *and* ν *is the largest number of edge-disjoint cycles contained in the demand graph* G^d.

Proof. First of all, the optimal solution requires $\mathsf{OPT} \leq D - \nu$ light paths, because it is straightforward to produce a feasible solution using $D - \nu$ light paths when given a collection of ν edge-disjoint cycles in G^d. We also show that $\mathsf{OPT} \geq D - \nu$ by induction on D. The base case $D = 1$ is trivial. For the induction hypothesis, suppose that the property holds for all $D < \bar{D}$, where $\bar{D}$ is an integer greater than 1. Then, if $D = \bar{D}$ we distinguish the following cases.

- If $\mathsf{OPT} \leq D - 2$, then the demand set $\mathcal{D}$ can be partitioned into non-empty subsets $\mathcal{D}_1$ and $\mathcal{D}_2$ such that (say) OPT_1 of those light paths serve demands in $\mathcal{D}_1$ only, and $\mathsf{OPT}_2 = \mathsf{OPT} - \mathsf{OPT}_1$ of them serve demands in $\mathcal{D}_2$ only. If ν_1 and ν_2 are the maximum number of edge-disjoint cycles in the graphs $(V, \mathcal{D}_1)$ and $(V, \mathcal{D}_2)$, respectively, then clearly $\nu_1 + \nu_2 \leq \nu$. Furthermore, by the induction hypothesis, $\mathsf{OPT}_1 \geq |\mathcal{D}_1| - \nu_1$ and $\mathsf{OPT}_2 \geq |\mathcal{D}_2| - \nu_2$, implying that $\mathsf{OPT} = \mathsf{OPT}_1 + \mathsf{OPT}_2 \geq D - \nu_1 - \nu_2 \geq D - \nu$.
- If $\mathsf{OPT} = D - 1$, then G^d cannot be acyclic. In other words, $\nu \geq 1$ and hence $\mathsf{OPT} \geq D - \nu$.
- If $\mathsf{OPT} = D$, then obviously $\mathsf{OPT} \geq D - \nu$, since $\nu \geq 0$.

Therefore, $\mathsf{OPT} = D - \nu$. □

Recall that the problem of finding a maximum-cardinality set of edge-disjoint cycles contained in an undirected multigraph is named *undirected cycle packing*. It is equivalent to light path minimization with half-wavelength demands in terms of polynomial-time reductions, as a direct consequence of Corollary 3.4. On the other hand, these two problems differ significantly with respect to approximation. Indeed, it is well-known that undirected cycle packing is quasi-NP-hard to approximate within a factor $\Omega\left(\log^{1/2-\epsilon} n\right)$ of the optimal [19], and the best known approximation algorithm for it attains a ratio of $O\left(\sqrt{\log n}\right)$ [17,18]. By contrast, for light path minimization with half-wavelength demands we present Algorithm 3.1, which has a constant ratio.

Theorem 3.5. *The approximation ratio of Algorithm 3.1 does not exceed* $\frac{1409}{1080} \approx 1.305$*, but it is at least* 1.304.

Proof. For our analysis, we consider a hypothetical collection Γ of edge-disjoint cycles in G^d. At first, Γ has the maximum possible cardinality: $|\Gamma| = \nu$. By Corollary 3.4, the optimal solution needs $\mathsf{OPT} = D - \nu = D - \sum_{i\geq 2} y_i$ light paths, where y_i indicates the (initial) number of i-cycles in Γ. In each iteration of Algorithm 3.1, Γ is modified as follows. Let C_{short} be the cycle extracted during the iteration in question. If C_{short} is in Γ, it is simply removed from that collection. Otherwise, C_{short} may have one or more common edges with each of $\tau \leq |C_{\text{short}}|$ cycles of Γ, which we denote by $C_1, \ldots, C_\tau$; note that possibly $\tau = 0$. If every edge of C_{short} belongs to one of $C_1, \ldots, C_\tau$, then the other edges of these cycles form a new cycle C', with $|C'| = \left(\sum_{j=1}^{\tau} |C_j|\right) - |C_{\text{short}}|$. In particular, the aforementioned condition is true if $\tau = |C_{\text{short}}|$. Thus, we remove $C_1, \ldots, C_\tau$ from Γ and (if applicable) add C' to it. In any case, observe that Γ becomes a collection of edge-disjoint cycles in the residual demand graph, i.e. what remains after the edges of C_{short} have been deleted.

It is straightforward to realize that when the algorithm extracts a 2-cycle, both $|\Gamma|$ and the number of 2-cycles in Γ are reduced by at most 1. Consequently, there exists a collection of ν disjoint cycles in G^d that contains all the 2-cycles extracted by Algorithm 3.1 — and, naturally, no other 2-cycles may belong to such a collection. Without loss of generality, we henceforth use that particular collection as Γ in our analysis.

Let $x_i, i \geq 2$, denote the number of i-cycles found and removed by the algorithm, and $\nu' = \sum_{i\geq 2} x_i$. By the above argument, $x_2 = y_2$. Since $\sum_{i\geq 2} y_i = \nu$ and $2y_2 + 3\sum_{i\geq 3} y_i \leq D$, we deduce that $x_2 = y_2 \geq 3\nu - D$. Extending this reasoning also yields the inequality $y_3 \geq 4\nu - D - 2x_2$. Moreover, every time Algorithm 3.1 extracts a 3-cycle, the number of 3-cycles of Γ decreases by at most 3; hence $x_3 \geq \frac{1}{3}y_3 \geq \frac{1}{3}(4\nu - D - 2x_2)$.

The solution produced by the algorithm requires $D - \nu'$ light paths. Assuming that OPT is a constant positive quantity, the linear program LP2 seeks to maximize the ratio of $D - \nu'$ over OPT under the above constraints. Its value is therefore an upper bound on the approximation factor achieved by the greedy algorithm, which is why it is called a *factor-revealing* linear program. It is easy to compute the optimal solution of LP2, whose value is $\frac{4}{3}$: $D = 3\mathsf{OPT}/2$, $\nu = \mathsf{OPT}/2$, $\nu' = \mathsf{OPT}/6$, $x_2 = y_2 = 0, x_3 = \mathsf{OPT}/6, y_3 = \mathsf{OPT}/2$, and $x_i = y_i = 0$ for $i \geq 4$.

Now let us sketch how to systematically refine LP2. One approach would be to add inequalities involving x_4, x_5, and so on, using similar arguments as for (2e). Note that such inequalities are based on how many cycles are *removed* from Γ in

$$\mathsf{LP2}: \quad \max \quad \big(D-\nu'\big)/\mathsf{OPT} \tag{2a}$$
$$\text{s.t.} \quad D-\nu = \mathsf{OPT} \tag{2b}$$
$$\sum_{i\geq 2} x_i = \nu' \tag{2c}$$
$$x_2 \geq 3\nu - D \tag{2d}$$
$$3x_3 \geq 4\nu - D - 2x_2 \tag{2e}$$
$$D, \nu, \nu', x_i \geq 0 \qquad \forall i \geq 2 \tag{2f}$$

$$\mathsf{LP3}: \quad \max \quad \big(D-\nu'\big)/\mathsf{OPT} \tag{3a}$$
$$\text{s.t.} \quad D-\nu = \mathsf{OPT} \tag{3b}$$
$$\sum_{i\geq 2} x_i = \nu' \tag{3c}$$
$$\sum_{i\geq 2} y_i = \nu \tag{3d}$$
$$x_2 = y_2 \tag{3e}$$
$$x_i = \sum_{\tau=1}^{i} \sum_{\substack{a_1,\ldots,a_\tau \\ i\leq a_1\leq\cdots\leq a_\tau}} z_i^{a_1,\ldots,a_\tau} \qquad \forall i \geq 3 \tag{3f}$$
$$\sum_{k=3}^{i}\sum_{\tau=1}^{k} \sum_{\substack{a_1,\ldots,a_\tau \\ k\leq a_1\leq\cdots\leq a_\tau}} \big|\{m \mid a_m = i\}\big|\, z_k^{a_1,\ldots,a_\tau} \geq y_i + \sum_{j=1}^{i/2} \sum_{\substack{a_1,\ldots,a_j \\ j\leq a_1\leq\cdots\leq a_j \\ a_1+\cdots+a_j=i-j}} z_j^{a_1,\ldots,a_j} \qquad \forall i \geq 3 \tag{3g}$$
$$D, \nu, \nu', x_i, y_i \geq 0 \qquad \forall i \geq 2 \tag{3h}$$
$$z_i^{a_1,\ldots,a_\tau} \geq 0 \qquad \forall i \geq 3,\ \forall 1 \leq \tau \leq i,\ \forall a_1,\ldots,a_\tau : i \leq a_1 \leq \cdots \leq a_\tau \tag{3i}$$

any given iteration, but do not account for the fact that a cycle may also be *added* to Γ at the same time. The latter is the focus of our refinement, for which we need to create new variables: for integers $i \geq 3$ and $\tau \geq 1$, with $\tau \leq i$, as well as integer indices $a_1, \ldots, a_\tau$ with $i \leq a_1 \leq \cdots \leq a_\tau$, let $z_i^{a_1,\ldots,a_\tau}$ denote the number of iterations of Algorithm 3.1 in which an i-cycle was extracted, having common edges with τ cycles of the then-current collection Γ, whose sizes are $a_1, \ldots, a_\tau$. In the extended linear program LP3, the inequalities (3g) effectively replace (2e) from LP2. The left-hand side of each of these inequalities indicates how many i-cycles are removed from Γ during the execution of Algorithm 3.1, whereas the right-hand side shows how many i-cycles were initially in Γ, plus a lower bound on the number of such cycles added to Γ during the execution of the algorithm.

To make the program tractable for an LP solver, we keep only those variables and inequalities needed to express the fact that when Algorithm 3.1 extracts a 3-cycle which overlaps with three 3-cycles of Γ, then these are removed and a new 6-cycle is added to Γ. This "trimmed" formulation suffices to obtain the claimed $\frac{1409}{1080}$ bound.

On the other hand, we recursively construct an infinite family of counterexamples $G^{d_0}, G^{d_1}, G^{d_2}, \ldots$ for which the above inequalities are essentially tight. First, we define an auxiliary integer sequence $\{\alpha_k\}$ such that $\alpha_1 = 3$ and $\alpha_k = \alpha_{k-1}^2 - \alpha_{k-1}$.

- The demand graph G^{d_0} is just a α_1-cycle, i.e. a 3-cycle.
- For $k \geq 1$, create α_k distinct copies of $G^{d_{k-1}}$: $G_1^{d_{k-1}}, \ldots, G_{\alpha_k}^{d_{k-1}}$. Additionally, let (u, v) be some arbitrarily chosen edge of $G^{d_{k-1}}$. For $j = 1$ to α_k, identify the copy of vertex v in $G_j^{d_{k-1}}$ with the copy of u in $G_{j+1}^{d_{k-1}}$ (index arithmetic is done modulo α_k), so that the copies of (u, v) form an α_k-cycle, which we call the *connecting cycle*. This constitutes the demand graph G^{d_k}.

Algorithm 3.2 Grooming for light path minimization with demands requesting arbitrary bitrates

$u \leftarrow \arg\max_{v \in V}\{t(v)\}$
for each $v \in V \setminus u$ **do**
  create $\left\lceil \frac{t(v)}{B} \right\rceil$ light paths along the shortest path from v to u
for each $d = (v_1, v_2) \in \mathcal{D}$ **do**
  route d through (at most) two light paths, one from v_1 to u and another from u to v_2

For every G^{d_k}, $k \geq 1$, there is a solution using exactly $2\prod_{j=1}^{k} \alpha_j$ light paths. In particular, it simply uses $\alpha_1 - 1 = 2$ light paths for each of the copies of G^{d_0} that together form G^{d_k}. By contrast, if Algorithm 3.1 happens to break ties in an unfavorable way, it produces a solution with at least

$$S_k = \left(3\prod_{j=1}^{k} \alpha_j\right) - 2 - \sum_{l=2}^{k}\prod_{m=l}^{k} \alpha_m \tag{4}$$

light paths. Take G^{d_1}, for example: it consists of α_1 copies of G^{d_0}, which are 3-cycles. Alternatively, the edges of G^{d_1} may be partitioned into one α_1-cycle, which is the connecting cycle of G^{d_1}, and one α_2-cycle. If Algorithm 3.1 first extracts the connecting cycle instead of another 3-cycle that is a copy G^{d_0}, then it has no choice but to extract the α_2-cycle afterwards. This leads to a solution with cost $(\alpha_1 - 1) + (\alpha_2 - 1) = 7 = S_2$.

Now, we make the inductive hypothesis that Algorithm 3.1 may produce a solution to instance $G^{d_{k'}}$ with at least $S_{k'}$ light paths, for some $k' \geq 1$, and that the last cycle extracted is a $\alpha_{k'+1}$-cycle. For $G^{d_{k'+1}}$, Algorithm 3.1 first extracts cycles from each of the $\alpha_{k'+1}$ copies of $G^{d_{k'}}$ until only an $\alpha_{k'+1}$-cycle remains from each copy. So far, at least $\alpha_{k'+1}(S_{k'} - \alpha_{k'+1} + 1)$ light paths have been used. The algorithm then extracts the $\alpha_{k'+1}$-cycle that is the connecting cycle of $G^{d_{k'+1}}$, and finally the remaining $\alpha_{k'+2}$-cycle. Therefore, the total number of light paths is at least

$$\alpha_{k'+1}(S_{k'} - \alpha_{k'+1} + 1) + (\alpha_{k'+1} - 1) + (\alpha_{k'+2} - 1) = S_{k'+1}.$$

This completes the inductive proof of (4). For $k = 1$ to 4, the approximation ratios are $\frac{7}{6}$, $\frac{23}{18}$, $\frac{176}{135}$, and $\frac{306457}{234900}$, with the latter already exceeding 1.304. □

3.3. Arbitrary demands

Although the structural results of the previous section do not apply for demands with arbitrary b_d, we provide a simple method (described in Algorithm 3.2) that yields a feasible solution with at most double the optimal value. In our notation, $t(v)$ denotes the total bitrate $\sum_d b_d$ of demands $d \in \mathcal{D}$ that have an endpoint at node $v \in V$.

Theorem 3.6. *The approximation ratio of Algorithm 3.2 is at most 2.*

Proof. Obviously, in any feasible solution there must be at least $\left\lceil \frac{t(v)}{B} \right\rceil$ light paths having an endpoint at any given node $v \in V$. Since every light path has two endpoints, $\frac{1}{2}\sum_{v \in V} \left\lceil \frac{t(v)}{B} \right\rceil$ is a lower bound on the value OPT of the optimal solution. On the other hand, Algorithm 3.2 uses $\sum_{v \in V \setminus u} \left\lceil \frac{t(v)}{B} \right\rceil$ light paths, which is at most twice OPT.

Note that this analysis would remain valid even if Algorithm 3.2 chose an arbitrary node of V as u. □

3.4. Connection with interchange distance in strings

The *interchange distance* $d_I(x, y)$ between two equal-length strings x, y over some alphabet Σ is the minimum number of interchanges required to transform x into y, or infinity if such a transformation is impossible. Here, *interchange* denotes the operation of swapping the positions of any two (not necessarily neighboring) elements of the string.

Determining the interchange distance $d_I(x, y)$ is feasible in linear time for *permutation strings*, i.e. strings whose elements are all distinct from one another [21]. In the general case, though, the problem is equivalent to computing the quantity $|E| - \nu$, where ν is the largest number of edge-disjoint cycles contained in a given *directed* Eulerian multigraph $G = (V, E)$. This was proven in [22]. Further, [22] showed that the problem admits a $\frac{3}{2}$-approximation algorithm, and also that its decision version is NP-hard. The latter result was based on a reduction from 3SAT to the problem of edge-partitioning an undirected graph into 3-cycles, due to [25]. As observed in [22], the edges of the reduction instance may be oriented without affecting its salient properties, thus extending its applicability to directed graphs. In summary,

Theorem 3.7 (*[22]*)**.** *The interchange distance problem in general strings is* NP*-hard. It has a* $\frac{3}{2}$*-approximation.*

We offer the following improvement to Theorem 3.7.

Theorem 3.8. *The interchange distance problem in general strings is* APX*-hard. It has a* 1.305*-approximation.*

Proof. Since the interchange distance problem is equivalent to computing $|E| - \nu$, where ν is the largest number of edge-disjoint cycles [22], we apply Algorithm 3.1 to approximately compute $|E| - \nu$ by repeatedly extracting the shortest *directed* cycle from G. Note that although Algorithm 3.1 and Theorem 3.5 are stated for undirected graphs, the analysis remains valid, because the claims therein are not affected by the directionality of the cycles. Theorem 3.5 immediately gives a 1.305 approximation.

We now prove the hardness of the interchange distance problem. Recall that Max 2SAT-3 is APX-hard [23,24], which means that there exists a constant $\epsilon > 0$ such that, given a Max 2SAT-3 instance ϕ with n clauses, it is NP-hard to distinguish whether at most $c \geq n/2$ clauses of ϕ are satisfiable, or at least $(1+\epsilon)c$ clauses are satisfiable. Note again the restriction $c \geq n/2$, which is due to the fact that at least half the clauses of ϕ are always satisfiable, e.g. by a simple greedy algorithm.

From such an instance ϕ, we construct a directed graph $G = (V, E)$ that does not contain 2-cycles, with $|E| = \delta n$ for a specific constant δ. The construction is identical to the one used for the NP-hardness reductions in [25,22]; it guarantees that if all clauses of ϕ are satisfiable, then $|E| - \nu = |E| - |E|/3 = 2|E|/3$ (because in that case G can be edge-partitioned into 3-cycles), or more generally that if a truth assignment satisfies $\geq s$ clauses of ϕ, then $|E| - \nu \leq 2|E|/3 + n - s$, and vice versa.

Therefore, if the value of $|E| - \nu$ could be approximated within a

$$\begin{aligned}\frac{2|E|/3 + n - c}{2|E|/3 + n - (1+\epsilon)c} &= 1 + \frac{\epsilon c}{(1 + 2\delta/3)n - (1+\epsilon)c} \\ &\geq 1 + \frac{\epsilon n/2}{(1 + 2\delta/3)n - (1+\epsilon)n/2} \\ &= 1 + \frac{\epsilon}{1 + 4\delta/3 - \epsilon}\end{aligned}$$

factor, then we would be able to distinguish whether at least $(1+\epsilon)c$ clauses are satisfiable or at most c are, which is NP-hard. We hence conclude that the interchange distance problem is APX-hard. □

4. Improved grooming with general costs in line and tree topologies

In this section we focus on improving the approximation ratios for grooming with general costs. More specifically, assuming half-wavelength demands, we present algorithms with small constant approximation ratios for line and tree topologies. Recall that for half-wavelength demands the problem is APX-hard, and for arbitrary demands the approximation ratio is logarithmic.

Recall also the wavelength-parsimonious constraint, which stipulates that in a feasible solution exactly $p(e) = \lceil \ell(e)/B \rceil$ light paths must use each link e. As a result, wavelength usage is kept at a minimum — at least with respect to the demand routing implied by the solution. Hence, such a concept is more useful for acyclic topologies, in which edge loads are fixed because the routing is unique. For the rest of the section, we refer to light path minimization under this constraint as *parsimonious grooming* and to the original problem, without the constraint, as *spendthrift grooming*, since in that case we do not care about wavelength usage at all.

In the following, we first develop algorithms for parsimonious grooming, and then show how they can be combined with Algorithm 3.1 to guarantee good approximations for half-wavelength instances of grooming with general costs.

4.1. Parsimonious grooming with half-wavelength demands

We begin with several concepts that play an important role in our algorithms. An instance is called *complete* if the load $\ell(e)$ is an integral multiple of the wavelength capacity B, for all $e \in E$. Otherwise, the instance is *incomplete*. In an incomplete instance, a *gap* is a maximal connected subgraph of G such that none of its edges' loads are integral multiples of B. *Augmentation* is the process of creating a complete instance from an incomplete one, by adding so-called *filler demands*.

Our positive results, presented below, are thus far applicable only to instances with half-wavelength demands. Unlike before, however, in parsimonious grooming the graph topology greatly impacts our ability to find good solutions, and consequently we distinguish several cases for which we develop specialized techniques. Henceforth, we say that a light path is *full* if it carries two half-wavelength demands, and *half-full* if it carries only one.

Line instances. Given an incomplete grooming instance with half-wavelength demands on a line, it can be augmented to a complete instance by covering each gap with one half-wavelength filler demand.

Lemma 4.1. *The augmentation process described above does not change the value of the optimal solution.*

Proof. Consider an optimal parsimonious solution for the incomplete instance, i.e. before it is augmented. For each gap, there must exist a collection of half-full light paths that covers the gap exactly. To see this, suppose we number the nodes of the line topology in order, so that gaps may be denoted by intervals, and consider a gap $[i, j]$. Since the optimal solution is parsimonious, all light paths that contain $[i, i+1]$ must be full, except one. This half-full light path must also terminate at i, because $[i-1, i]$ is not part of the gap and thus all light paths that contain $[i-1, i]$ must be full. Let i' be the other

Algorithm 4.1 Parsimonious grooming with half-wavelength demands on a line

```
if the instance is incomplete then
  for each gap do
    {v1, v2} ← endpoints of the gap
    add a filler demand (v1, v2) to the instance
while the current demand graph is not acyclic do
  C_short ← the shortest cycle in the demand graph
  use |C_short| − 1 light paths to serve the demands in C_short
  remove C_short from the demand graph
```

endpoint of that light path; clearly $i' \leq j$. If $i' < j$, then we can then repeat the above argument on the remaining gap $[i', j]$ to identify additional half-full light paths. All these light paths are consecutive, and their union covers the gap exactly. Therefore, when the filler demand for gap $[i, j]$ is added, it can be served by said light paths without increasing the value of the optimal solution. □

Let G^{d^*} and D^* denote the demand graph and the number of demands of the resulting complete instance, respectively. A simple variant of Algorithm 3.1 may now be used to obtain a solution. Algorithm 4.1 describes the entire process, including augmentation. Note that G^{d^*} is Eulerian, so after the algorithm greedily removes as many cycles as it can, the remaining graph is empty. Hence, the statement and proof of Theorem 3.5 carry over to Algorithm 4.1 as well.

Theorem 4.2. *The approximation ratio of Algorithm 4.1 does not exceed* $\frac{1409}{1080} \approx 1.305$.

Tree instances. We augment an incomplete instance on a tree as follows. If a gap is a path, in the graph-theoretic sense, we create one filler demand for the gap; if a gap is a tree, we partition it into a collection of paths with properties stated in the following lemma.

Lemma 4.3. *A tree T can be efficiently partitioned into internally disjoint paths, such that no two of these paths have a common endpoint. The number of these paths equals half the number of odd-degree nodes in T.*

Proof. We greedily partition the tree T into paths by repeatedly choosing a path whose endpoints have degree 1 in T, then removing its edges from T and recursing on the residual graph. Since T is acyclic, this procedure will continue until the graph becomes empty. Note that the endpoints of each path thus created are odd-degree nodes of (the original) T, and every such node is an endpoint of exactly one path. □

Let G^{d^*} and D^* denote the demand graph and the number of demands of the resulting complete instance, respectively. In contrast to Lemma 4.1, now we do not know whether the above augmentation would increase the value of the optimal parsimonious solution. Instead, we show the following.

Lemma 4.4. *There exists an augmentation that produces an instance with D^* demands, without affecting the optimal value.*

Proof. Consider an optimal parsimonious solution for the incomplete instance before it is augmented. For each gap, there must exist a collection of half-full light paths such that the physical routes of these light paths cover the gap exactly. To see this, we start with any degree-1 node v within a gap. As in the proof of Lemma 4.1, there must exist a half-full light path that terminates at v. We create one filler demand for this light path, and repeat the argument on the remaining gap. Moreover, if two such filler demands share a common endpoint, then we concatenate them into one. Obviously, augmenting with these filler demands preserves the optimal value. On top of that, it is easy to see that their number equals half the number of odd-degree nodes of the gap. Consequently, the overall augmentation results in a complete instance with exactly D^* demands. □

Algorithm 4.2 shows how to obtain a feasible solution for any given instance, whether complete or incomplete. Its correctness is deduced from the following lemma.

Lemma 4.5. *Provided the demand graph is non-empty, at least one of the conditions in lines 13 and 16 of Algorithm 4.2 is always satisfied.*

Proof. For simplicity, we refer to these conditions as Condition A and Condition B, respectively. Suppose that between lines 10 and 11 of the algorithm we add a call to Procedure 4.3, which constructs a walk w in G. By that point, it is ensured that the instance is complete, via augmentation if necessary. Thus, due to parity, the assignments in lines 4 and 8 of the procedure are always valid. In every iteration, w is extended by concatenating it with the path representing d_{next}. However, for line 14 to be executed, d_{next} must not touch any point common to d_{curr} and d_{prev} (including v_1), and must not be entirely contained in d_{curr}. Hence, since G is acyclic, w covers at least one new edge in every iteration, implying that the procedure eventually terminates and consequently either Condition A or B holds. □

Theorem 4.6. *The approximation ratio of Algorithm 4.2 is at most* 4.

Algorithm 4.2 Parsimonious grooming with half-wavelength demands on a tree

1: **if** the instance is incomplete **then**
2: **for** each gap that is a path **do**
3: $\{v_1, v_2\} \leftarrow$ endpoints of the gap
4: add a filler demand (v_1, v_2) to the instance
5: **for** each gap that is a tree **do**
6: greedily partition the gap into paths, according to Lemma 4.3
7: **for** each path in the decomposition **do**
8: $\{v_1, v_2\} \leftarrow$ endpoints of the path
9: add a filler demand (v_1, v_2) to the instance
10: **while** the current demand graph is non-empty **do**
11: **if** there exists a 2-cycle C_2 in the demand graph **then**
12: use one light path to serve the demands in C_2; remove C_2 from the demand graph
13: **else if** for some $v, u_1, u_2 \in V$ there exist demands $d_1 = (v, u_1)$ and $d_2 = (v, u_2)$ in the demand graph, and u_1 lies on the unique path from v to u_2 in G **then**
14: create a light path from v to u_1, in order to serve d_1 and (partially) d_2
15: remove d_1 from the demand graph; truncate d_2 to (u_1, u_2)
16: **else if** for some $v, u_1, u_2, w_1, w_2 \in V$ there exist demands $d_0 = (u_1, u_2)$, $d_1 = (u_1, w_1)$, and $d_2 = (u_2, w_2)$ in the demand graph, and v lies in the following three unique paths in G: (i) from u_1 to u_2, (ii) from u_1 to w_1, and (iii) from u_2 to w_2 **then**
17: create two light paths, from v to u_1 and to u_2, respectively, in order to serve d_0 and (partially) d_1 and d_2
18: remove d_0 from the demand graph; truncate d_1 to (v, w_1) and d_2 to (v, w_2)

Procedure 4.3 Constructing a walk in the tree G

1: $d_{\text{curr}} \leftarrow$ a randomly chosen demand
2: $\{v_1, v_2\} \leftarrow$ endpoints of d_{curr}
3: $w \leftarrow$ the unique path between v_1 and v_2 in G
4: $d_{\text{prev}} \leftarrow$ a demand other than d_{curr} with endpoint at v_1
5: **if** Condition A holds for d_{curr} and d_{prev} **then**
6: **return** w
7: **loop**
8: $d_{\text{next}} \leftarrow$ a demand other than d_{curr} with endpoint at v_2
9: $v_3 \leftarrow$ the endpoint of d_{next} other than v_2
10: **if** Condition A holds for d_{curr} and d_{next} **then**
11: **return** w
12: **if** Condition B holds for d_{curr}, d_{prev} and d_{next} **then**
13: **return** w
14: append to w the unique path from v_2 to v_3 in G
15: $v_1 \leftarrow v_2$; $v_2 \leftarrow v_3$; $d_{\text{prev}} \leftarrow d_{\text{curr}}$; $d_{\text{curr}} \leftarrow d_{\text{next}}$

Proof. By adding filler demands Algorithm 4.2 produces a complete instance with exactly D^* demands. Consider the three cases in the while-loop in Algorithm 4.2. In case 1 (line 11), two demands are removed and one light path is created. In case 2 (line 13), one demand is removed, another is truncated, and one light path is created. In case 3 (line 16), one demand is removed and two others are truncated, while two light paths are created. Therefore, for each removed demand at most two new light paths are added, so Algorithm 4.2 yields a solution of at most $2D^*$ light paths.

Finally, the augmentation from Lemma 4.4 implies that the optimal value is at least $\frac{D^*}{2}$, since every light path can serve at most 2 of the D^* half-wavelength demands of the resulting augmented instance. This completes our proof. □

4.2. Grooming with general costs

Spendthrift and parsimonious grooming are at opposite extremes in terms of wavelength usage cost. Now, we shall utilize both to construct an algorithm for grooming with general costs:

- treat the instance as an instance of spendthrift grooming and obtain a solution S;
- treat the instance as an instance of parsimonious grooming and obtain a solution P;
- compare S and P with respect to their overall cost (endpoint equipment plus wavelength usage) and keep the less expensive one as the final solution.

Given a solution X, let path(X) be first term of (1), i.e. the number of light paths in X, and let wave(X) be the second term of (1), i.e. the wavelength usage cost of X. Additionally, let O_s, O_p and O_g be the optimal solutions with respect to spendthrift grooming, parsimonious grooming and grooming with general costs, respectively.

To obtain the spendthrift solution S we use Algorithm 3.1, with the following adjustment: each light path used by the algorithm must follow the shortest physical route in G, with respect to $c(e)$. Clearly, this adjustment is straightforward to implement, but it also ensures the following.

Lemma 4.7. *For half-wavelength demands, the wavelength usage cost of the solution S,* wave(S)*, is at most twice the minimum possible.*

Proof. Let $L = \sum_{d\in\mathcal{D}} \text{dist}(d)$, where dist($d$) is the shortest path distance, with respect to $c(e)$, between the endpoints of d. The optimal wavelength usage cost is obviously at least $L/2$. Furthermore, each light path used by Algorithm 3.1 serves a demand that is not served by any other light path. Thus, the algorithm's wavelength usage cost is at most L. □

Likewise, we need to show that algorithms for parsimonious grooming yield solutions with reasonably low endpoint equipment cost.

Lemma 4.8. *Given a line (or tree) instance, the ratio of light path counts under the parsimonious and optimal spendthrift solutions* $\frac{\text{path}(P)}{\text{path}(O_s)}$ *does not exceed* 2 *(or* 4*).*

Proof. The proof of Theorem 4.6 shows that, for a tree, $\text{path}(P) \le 2D^*$ and $\text{path}(O_s) \ge D^*/2$. Hence, $\frac{\text{path}(P)}{\text{path}(O_s)} \le 4$. Similarly, Algorithm 4.1 guarantees $\text{path}(P) \le D^*$ for the line instance. As a result, $\frac{\text{path}(P)}{\text{path}(O_s)} \le 2$. □

Lemmas 4.7 and 4.8 imply that both solutions S and P, as defined above, have near-optimal overall cost. Nevertheless, by retaining only the less expensive solution we can guarantee a better approximation ratio.

Theorem 4.9. *The aforementioned algorithm guarantees approximation ratio at most* 1.590 *(or* 1.812*) for line (or tree) instances of grooming with general costs and half-wavelength demands.*

Proof. Lemma 4.7 implies that $\text{wave}(S) \le 2 \cdot \text{wave}(O_g)$. By definition, $\text{path}(O_s) \le \text{path}(O_g)$ and, for acyclic topologies, $\text{wave}(P) = \text{wave}(O_p) \le \text{wave}(O_g)$. Moreover, $\text{path}(S) \le 1.305 \cdot \text{path}(O_s)$ by Theorem 3.5. For the line case, Lemma 4.8 implies that $\text{path}(P) \le 2 \cdot \text{path}(O_s)$. Therefore, the spendthrift solution S has overall cost at most $1.305 \cdot \text{path}(O_g) + 2 \cdot \text{wave}(O_g)$ and the parsimonious solution P has overall cost at most $2 \cdot \text{path}(O_g) + \text{wave}(O_g)$. Choosing the best of the two solutions implies an approximation ratio of

$$\frac{\min\{1.305 \cdot \text{path}(O_g) + 2 \cdot \text{wave}(O_g),\ 2 \cdot \text{path}(O_g) + \text{wave}(O_g)\}}{\text{path}(O_g) + \text{wave}(O_g)}$$

for the line topology. The worst case occurs when the two quantities in the numerator are equal, i.e. when $\frac{\text{wave}(O_g)}{\text{path}(O_g)} = 0.695$, so this ratio is bounded by

$$\frac{1.305 + 2 \times 0.695}{1 + 0.695} \le 1.590.$$

Similar analysis yields an approximation ratio of

$$\frac{\min\{1.305 \cdot \text{path}(O_g) + 2 \cdot \text{wave}(O_g),\ 4 \cdot \text{path}(O_g) + \text{wave}(O_g)\}}{\text{path}(O_g) + \text{wave}(O_g)}$$

for the tree topology, which does not exceed

$$\frac{1.305 + 2 \times 2.695}{1 + 2.695} \le 1.812.$$ □

5. Conclusions and open problems

In this paper, we studied traffic grooming for optical network design. We described two related optimization objectives, each having its own significance, and proposed simple approximation algorithms for arbitrary traffic demands on arbitrary network topologies. Additionally, we developed specialized techniques that offer improved approximation guarantees in more restricted settings, such as half-wavelength demands, and established that even those special cases are hard to approximate arbitrarily closely.

There remain a few interesting open questions on grooming from the theoretical perspective. In particular, developing an algorithm with a non-trivial approximation ratio for parsimonious grooming with arbitrary demands appears to be challenging, even on path instances. Furthermore, establishing inapproximability results for the aforementioned version of grooming would complement our understanding of the problem.

Another direction for future research, inspired by [9], is to investigate the dual problem of maximizing the number of satisfied demands, subject to limitations on available endpoint equipment. Apart from the obvious scenario of optimally configuring a pre-existing network, certain technological considerations also motivate such an endeavor. In particular, each packet router occupies one special slot on an Optical Add/Drop Multiplexer, and each OADM provides a fixed number of such slots. Since OADMs are substantially more complex and expensive than packet routers, we are often obliged to take into account the constraints that the former impose on the number of the latter.

Acknowledgements

The first author's work was partially done at Columbia University, supported by NSF grant CCF-0728736 and an Alexander S. Onassis Foundation Scholarship.

References

[1] R. Dutta, G.N. Rouskas, Traffic grooming in WDM networks: past and future, IEEE Network 16 (6) (2002) 46–56.
[2] K. Zhu, B. Mukherjee, A review of traffic grooming in WDM optical networks: architectures and challenges, Optical Networks Magazine 4 (2) (2003) 55–64.
[3] J.-C. Bermond, D. Coudert, Traffic grooming in unidirectional WDM ring networks using design theory, in: Proceedings of IEEE ICC, 2003, pp. 1402–1406.
[4] N. Brauner, Y. Crama, G. Finke, P. Lemaire, C. Wynants, Approximation algorithms for the design of SDH/SONET networks, RAIRO Operations Research 37 (2003) 235–247.
[5] O. Goldschmidt, D.S. Hochbaum, A. Levin, E.V. Olinick, The SONET edge-partition problem, Networks 41 (1) (2003) 13–23.
[6] Y. Wang, Q.-P. Gu, Grooming of symmetric traffic in unidirectional SONET/WDM rings, in: Proceedings of IEEE ICC, 2006, pp. 2407–2414.
[7] M. Shalom, S. Zaks, A10/7 $+ \epsilon$ approximation for minimizing the number of ADMs in SONET rings, IEEE/ACM Transactions on Networking 15 (6) (2007) 1593–1602.
[8] L. Epstein, A. Levin, Better bounds for minimizing SONET ADMs, Journal of Computer and System Sciences 75 (2) (2009) 122–136.
[9] Y. Wang, Q.-P. Gu, Maximizing throughput for traffic grooming with limited grooming resources, in: Proceedings of IEEE GLOBECOM, 2007, pp. 2337–2341.
[10] O. Amini, S. Perennes, I. Sau, Hardness and approximation of traffic grooming, Theoretical Computer Science 410 (38–40) (2009) 3751–3760.
[11] S. Huang, R. Dutta, G.N. Rouskas, Traffic grooming in path, star, and tree networks: complexity, bounds and algorithms, IEEE Journal on Selected Areas in Communications 24 (4) (2006) 66–82.
[12] B. Awerbuch, Y. Azar, Buy-at-bulk network design, in: Proceedings of IEEE FOCS, 1997, pp. 542–547.
[13] M. Andrews, Hardness of buy-at-bulk network design, in: Proceedings of IEEE FOCS, 2004, pp. 115–124.
[14] M. Andrews, L. Zhang, Bounds on fiber minimization in optical networks with fixed fiber capacity, in: Proceedings of IEEE INFOCOM, 2005, pp. 409–419.
[15] A. Caprara, Sorting permutations by reversals and Eulerian cycle decompositions, SIAM Journal on Discrete Mathematics 12 (1) (1999) 91–110.
[16] A. Caprara, A. Panconesi, R. Rizzi, Packing cycles in undirected graphs, Journal of Algorithms 48 (2003) 239–256.
[17] M. Krivelevich, Z. Nutov, R. Yuster, Approximation algorithms for cycle packing problems, in: Proceedings of ACM–SIAM SODA, 2005, pp. 556–561.
[18] M. Krivelevich, Z. Nutov, M.R. Salavatipour, J. Verstraete, R. Yuster, Approximation algorithms and hardness results for cycle packing problems, ACM Transactions on Algorithms 3 (4) (2007).
[19] Z. Friggstad, M.R. Salavatipour, Approximability of packing disjoint cycles, in: Proceedings of ISAAC, 2007, pp. 304–315.
[20] A. Cayley, Note on the theory of permutations, Philosophical Magazine 34 (1849) 527–529.
[21] A. Amir, Y. Aumann, G. Benson, A. Levy, O. Lipsky, E. Porat, S. Skiena, U. Vishne, Pattern matching with address errors: rearrangement distances, in: Proceedings of ACM–SIAM SODA, 2006, pp. 1221–1229.
[22] A. Amir, T. Hartman, O. Kapah, A. Levy, E. Porat, On the cost of interchange rearrangement in strings, in: Proceedings of ESA, 2007, pp. 99–110.
[23] C.H. Papadimitriou, M. Yannakakis, Optimization, approximation, and complexity classes, Journal of Computer and System Sciences 43 (1991) 425–440.
[24] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti Spaccamela, M. Protasi, Complexity and Approximation: Combinatorial Optimization Problems and their Approximability Properties, Springer-Verlag, Berlin, 1999.
[25] I. Holyer, The NP-completeness of some edge-partition problems, SIAM Journal on Computing 10 (4) (1981) 713–717.